Foundation of Cryptography, Lecture 6
Interactive Proofs and Zero Knowledge

Iftach Haitner, Tel Aviv University

Tel Aviv University.

April 23, 2014

Part I

Interactive Proofs

$\mathcal{NP}$ as a Non-interactive Proofs

Definition 1 ($\mathcal{NP}$)

$\mathcal{L} \in \mathcal{NP}$ iff $\exists$ and poly-time algorithm V such that:

- $\forall x \in \mathcal{L} \cap \{0,1\}^n$ there exists $w \in \{0,1\}^*$ s.t. $\mathsf{V}(x,w) = 1$
- $\mathsf{V}(x,w) = 0$ for every $x \notin \mathcal{L}$ and $w \in \{0,1\}^*$

Only $|x|$ counts for the running time of V.

$\mathcal{NP}$ as a Non-interactive Proofs

Definition 1 ($\mathcal{NP}$)

$\mathcal{L} \in \mathcal{NP}$ iff $\exists$ and poly-time algorithm V such that:

- $\forall x \in \mathcal{L} \cap \{0,1\}^n$ there exists $w \in \{0,1\}^*$ s.t. $\mathsf{V}(x,w) = 1$
- $\mathsf{V}(x,w) = 0$ for every $x \notin \mathcal{L}$ and $w \in \{0,1\}^*$

Only $|x|$ counts for the running time of V.

A proof system

$\mathcal{NP}$ as a Non-interactive Proofs

Definition 1 ($\mathcal{NP}$)

$\mathcal{L} \in \mathcal{NP}$ iff $\exists$ and poly-time algorithm V such that:

- $\forall x \in \mathcal{L} \cap \{0,1\}^n$ there exists $w \in \{0,1\}^*$ s.t. $\mathsf{V}(x,w) = 1$
- $\mathsf{V}(x,w) = 0$ for every $x \notin \mathcal{L}$ and $w \in \{0,1\}^*$

Only $|x|$ counts for the running time of V.

A proof system

- Efficient verifier, efficient prover (given the witness)

$\mathcal{NP}$ as a Non-interactive Proofs

Definition 1 ($\mathcal{NP}$)

$\mathcal{L} \in \mathcal{NP}$ iff $\exists$ and poly-time algorithm V such that:

- $\forall x \in \mathcal{L} \cap \{0,1\}^n$ there exists $w \in \{0,1\}^*$ s.t. $\mathsf{V}(x, w) = 1$
- $\mathsf{V}(x, w) = 0$ for every $x \notin \mathcal{L}$ and $w \in \{0,1\}^*$

Only $|x|$ counts for the running time of V.

A proof system

- Efficient verifier, efficient prover (given the witness)
- Soundness holds unconditionally

Interactive proofs

Protocols between efficient verifier and unbounded provers.

Interactive proofs

Protocols between efficient verifier and unbounded provers.

Definition 2 (Interactive proof)

A protocol (P,V) is an interactive proof for $\mathcal{L}$, if V is PPT and:

Completeness $\forall x \in \mathcal{L}$, $\Pr[\langle(\mathsf{P},\mathsf{V})(x)\rangle = 1] \geq 2/3$

Soundness $\forall x \notin \mathcal{L}$, and any algorithm P^* $\Pr[\langle(\mathsf{P}^*,\mathsf{V})(x)\rangle = 1] \leq 1/3$

IP is the class of languages that have interactive proofs.

Interactive proofs

Protocols between efficient verifier and unbounded provers.

Definition 2 (Interactive proof)

A protocol (P, V) is an interactive proof for $\mathcal{L}$, if V is PPT and:

Completeness $\forall x \in \mathcal{L}$, $\Pr[\langle(\mathsf{P}, \mathsf{V})(x)\rangle = 1] \geq 2/3$

Soundness $\forall x \notin \mathcal{L}$, and any algorithm P^* $\Pr[\langle(\mathsf{P}^*, \mathsf{V})(x)\rangle = 1] \leq 1/3$

IP is the class of languages that have interactive proofs.

- $\mathsf{IP} = \mathrm{PSPACE}$!

Interactive proofs

Protocols between efficient verifier and unbounded provers.

Definition 2 (Interactive proof)

A protocol (P, V) is an interactive proof for $\mathcal{L}$, if V is PPT and:

Completeness $\forall x \in \mathcal{L}$, $\Pr[\langle(\mathsf{P}, \mathsf{V})(x)\rangle = 1] \geq 2/3$

Soundness $\forall x \notin \mathcal{L}$, and any algorithm P^* $\Pr[\langle(\mathsf{P}^*, \mathsf{V})(x)\rangle = 1] \leq 1/3$

IP is the class of languages that have interactive proofs.

- $\mathsf{IP} = \mathrm{PSPACE}$!
- We typically consider (and achieve) perfect completeness.

Interactive proofs

Protocols between efficient verifier and unbounded provers.

Definition 2 (Interactive proof)

A protocol (P, V) is an interactive proof for $\mathcal{L}$, if V is PPT and:

Completeness $\forall x \in \mathcal{L}$, $\Pr[\langle(\mathsf{P}, \mathsf{V})(x)\rangle = 1] \geq 2/3$

Soundness $\forall x \notin \mathcal{L}$, and any algorithm P^* $\Pr[\langle(\mathsf{P}^*, \mathsf{V})(x)\rangle = 1] \leq 1/3$

IP is the class of languages that have interactive proofs.

- $\mathsf{IP} = \mathrm{PSPACE}$!
- We typically consider (and achieve) perfect completeness.
- Negligible “soundness error" achieved via repetition.

Interactive proofs

Protocols between efficient verifier and unbounded provers.

Definition 2 (Interactive proof)

A protocol (P, V) is an interactive proof for $\mathcal{L}$, if V is PPT and:

Completeness $\forall x \in \mathcal{L}$, $\Pr[\langle(\mathsf{P}, \mathsf{V})(x)\rangle = 1] \geq 2/3$

Soundness $\forall x \notin \mathcal{L}$, and any algorithm P^* $\Pr[\langle(\mathsf{P}^*, \mathsf{V})(x)\rangle = 1] \leq 1/3$

IP is the class of languages that have interactive proofs.

- $\mathsf{IP} = \mathrm{PSPACE}$!
- We typically consider (and achieve) perfect completeness.
- Negligible “soundness error" achieved via repetition.
- Sometime we have efficient provers via “auxiliary input".

Interactive proofs

Protocols between efficient verifier and unbounded provers.

Definition 2 (Interactive proof)

A protocol (P,V) is an interactive proof for $\mathcal{L}$, if V is PPT and:

Completeness $\forall x \in \mathcal{L}$, $\Pr[\langle(\mathsf{P},\mathsf{V})(x)\rangle = 1] \geq 2/3$

Soundness $\forall x \notin \mathcal{L}$, and any algorithm P^* $\Pr[\langle(\mathsf{P}^*,\mathsf{V})(x)\rangle = 1] \leq 1/3$

IP is the class of languages that have interactive proofs.

- $\mathrm{IP} = \mathrm{PSPACE}$!
- We typically consider (and achieve) perfect completeness.
- Negligible “soundness error" achieved via repetition.
- Sometime we have efficient provers via “auxiliary input".
- Relaxation: *Computationally sound proofs* [also known as, *interactive arguments*]: soundness only guaranteed against efficient (PPT) provers.

Section 1

Interactive Proof for Graph Non-Isomorphism

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $\mathsf{G}_0 = ([m], E_0)$ and $\mathsf{G}_1 = ([m], E_1)$ are isomorphic, denoted $\mathsf{G}_0 \equiv \mathsf{G}_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $G_0 = ([m], E_0)$ and $G_1 = ([m], E_1)$ are isomorphic, denoted $G_0 \equiv G_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

- We assume a reasonable mapping from graphs to strings

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $G_0 = ([m], E_0)$ and $G_1 = ([m], E_1)$ are isomorphic, denoted $G_0 \equiv G_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

- We assume a reasonable mapping from graphs to strings
- graph isomorphism is an equivalence class.

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $\mathsf{G}_0 = ([m], E_0)$ and $\mathsf{G}_1 = ([m], E_1)$ are isomorphic, denoted $\mathsf{G}_0 \equiv \mathsf{G}_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

- We assume a reasonable mapping from graphs to strings
- graph isomorphism is an equivalence class.
- $\mathcal{GI} = \{(\mathsf{G}_0, \mathsf{G}_1) \colon \mathsf{G}_0 \equiv \mathsf{G}_1\} \in \mathcal{NP}$

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $\mathsf{G}_0 = ([m], E_0)$ and $\mathsf{G}_1 = ([m], E_1)$ are isomorphic, denoted $\mathsf{G}_0 \equiv \mathsf{G}_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

- We assume a reasonable mapping from graphs to strings
- graph isomorphism is an equivalence class.
- $\mathcal{GI} = \{(\mathsf{G}_0, \mathsf{G}_1) \colon \mathsf{G}_0 \equiv \mathsf{G}_1\} \in \mathcal{NP}$
- Does $\mathcal{GNI} = \{(\mathsf{G}_0, \mathsf{G}_1) \colon \mathsf{G}_0 \not\equiv \mathsf{G}_1\} \in \mathcal{NP}$?

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $\mathsf{G}_0 = ([m], E_0)$ and $\mathsf{G}_1 = ([m], E_1)$ are isomorphic, denoted $\mathsf{G}_0 \equiv \mathsf{G}_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

- We assume a reasonable mapping from graphs to strings
- graph isomorphism is an equivalence class.
- $\mathcal{GI} = \{(\mathsf{G}_0, \mathsf{G}_1) \colon \mathsf{G}_0 \equiv \mathsf{G}_1\} \in \mathcal{NP}$
- Does $\mathcal{GNI} = \{(\mathsf{G}_0, \mathsf{G}_1) \colon \mathsf{G}_0 \not\equiv \mathsf{G}_1\} \in \mathcal{NP}$?
- We will show a simple interactive proof for $\mathcal{GNI}$

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $G_0 = ([m], E_0)$ and $G_1 = ([m], E_1)$ are isomorphic, denoted $G_0 \equiv G_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

- We assume a reasonable mapping from graphs to strings
- graph isomorphism is an equivalence class.
- $\mathcal{GI} = \{(G_0, G_1) \colon G_0 \equiv G_1\} \in \mathcal{NP}$
- Does $\mathcal{GNI} = \{(G_0, G_1) \colon G_0 \not\equiv G_1\} \in \mathcal{NP}$?
- We will show a simple interactive proof for $\mathcal{GNI}$

Graph isomorphism

Π_m – the set of all permutations from $[m]$ to $[m]$

Definition 3 (graph isomorphism)

Graphs $G_0 = ([m], E_0)$ and $G_1 = ([m], E_1)$ are isomorphic, denoted $G_0 \equiv G_1$, if $\exists \pi \in \Pi_m$ such that
$(u, v) \in E_0$ iff $(\pi(u), \pi(v)) \in E_1$.

- We assume a reasonable mapping from graphs to strings
- graph isomorphism is an equivalence class.
- $\mathcal{GI} = \{(G_0, G_1) \colon G_0 \equiv G_1\} \in \mathcal{NP}$
- Does $\mathcal{GNI} = \{(G_0, G_1) \colon G_0 \not\equiv G_1\} \in \mathcal{NP}$?
- We will show a simple interactive proof for $\mathcal{GNI}$
  Idea: Beer tasting...

Interactive proof for $\mathcal{GNI}$

Protocol 4 ((P, V))

Common input $\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1)$

1. V chooses $b \leftarrow \{0, 1\}$ and $\pi \leftarrow \Pi_m$, and sends $\pi(E_b)$ to P.[a]
2. P send b' to V (tries to set $b' = b$).
3. V accepts iff $b' = b$.

[a] $\pi(E) = \{(\pi(u), \pi(v) \colon (u, v) \in E\}$.

Interactive proof for $\mathcal{GNI}$

Protocol 4 ((P, V))

Common input $G_0 = ([m], E_0), G_1 = ([m], E_1)$

1. V chooses $b \leftarrow \{0,1\}$ and $\pi \leftarrow \Pi_m$, and sends $\pi(E_b)$ to P.[a]
2. P send b' to V (tries to set $b' = b$).
3. V accepts iff $b' = b$.

[a] $\pi(E) = \{(\pi(u), \pi(v) \colon (u, v) \in E\}$.

Claim 5

The above protocol is IP for $\mathcal{GNI}$, with perfect completeness and soundness error $\frac{1}{2}$.

Proving Claim 5

- Graph isomorphism is an equivalence relation (separates the set of all graph pairs into separate subsets)

Proving Claim 5

- Graph isomorphism is an equivalence relation (separates the set of all graph pairs into separate subsets)

Proving Claim 5

- Graph isomorphism is an equivalence relation (separates the set of all graph pairs into separate subsets)
- $([m], \pi(E_i))$ is a random element in $[\mathsf{G}_i]$ — the equivalence class of G_i

Proving Claim 5

- Graph isomorphism is an equivalence relation (separates the set of all graph pairs into separate subsets)
- $([m], \pi(E_i))$ is a random element in $[\mathsf{G}_i]$ — the equivalence class of G_i

Proving Claim 5

- Graph isomorphism is an equivalence relation (separates the set of all graph pairs into separate subsets)
- $([m], \pi(E_i))$ is a random element in $[\mathrm{G}_i]$ — the equivalence class of G_i

Hence,

$\mathrm{G}_0 \equiv \mathrm{G}_1$: $\Pr[b' = b] \leq \frac{1}{2}$.

Proving Claim 5

- Graph isomorphism is an equivalence relation (separates the set of all graph pairs into separate subsets)
- $([m], \pi(E_i))$ is a random element in $[\mathrm{G}_i]$ — the equivalence class of G_i

Hence,

$\mathrm{G}_0 \equiv \mathrm{G}_1$: $\Pr[b' = b] \leq \frac{1}{2}$.

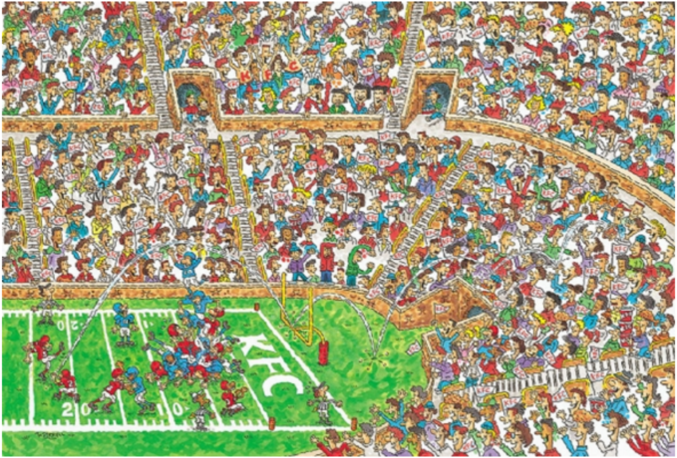
$\mathrm{G}_0 \not\equiv \mathrm{G}_1$: $\Pr[b' = b] = 1$ (i.e., P can, possibly inefficiently, extracted from $\pi(E_i)$)

□

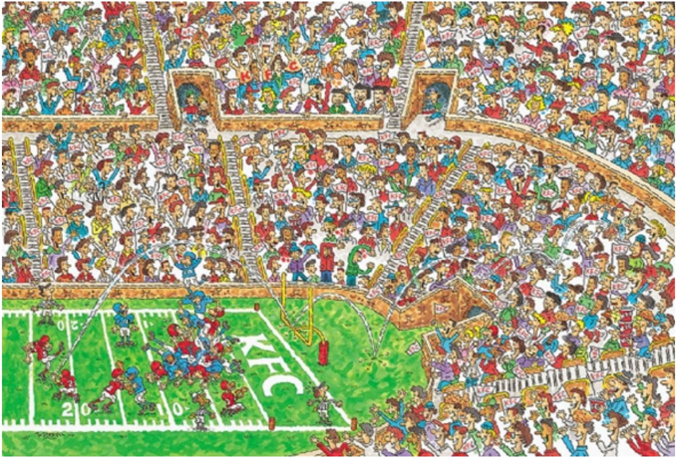
Part II

Zero knowledge Proofs

Where is Waldo?



Where is Waldo?



Question 6

Can you prove you know where Waldo is without revealing his location?

The concept of zero knowledge

- Proving w/o revealing any addition information.

The concept of zero knowledge

- Proving w/o revealing any addition information.
- What does it mean?

The concept of zero knowledge

- Proving w/o revealing any addition information.
- What does it mean?

  Simulation paradigm.

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P, V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P}, \mathsf{V}^*)(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$.

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P,V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P},\mathsf{V}^*)(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P,V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P},\mathsf{V}^*)(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

1. $\mathcal{ZK}$ is a property of the prover.

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P,V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P},\mathsf{V}^*)(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

1. $\mathcal{ZK}$ is a property of the prover.
2. $\mathcal{ZK}$ only required to hold with respect to true statements.

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P,V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P},\mathsf{V}^*)(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

1. $\mathcal{ZK}$ is a property of the prover.
2. $\mathcal{ZK}$ only required to hold with respect to true statements.
3. wlg. V^*'s outputs is its “view".

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P, V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P}, \mathsf{V}^*)(x)\rangle\}_{x \in \mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x \in \mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

1. $\mathcal{ZK}$ is a property of the prover.
2. $\mathcal{ZK}$ only required to hold with respect to true statements.
3. wlg. V^*'s outputs is its “view".
4. Trivial to achieve for $\mathcal{L} \in \mathcal{BPP}$

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P, V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P}, \mathsf{V}^*)(x)\rangle\}_{x \in \mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x \in \mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

1. $\mathcal{ZK}$ is a property of the prover.
2. $\mathcal{ZK}$ only required to hold with respect to true statements.
3. wlg. V^*'s outputs is its “view".
4. Trivial to achieve for $\mathcal{L} \in \mathcal{BPP}$
5. Extension: auxiliary input

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P, V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P}, \mathsf{V}^*)(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

1. $\mathcal{ZK}$ is a property of the prover.
2. $\mathcal{ZK}$ only required to hold with respect to true statements.
3. wlg. V^*'s outputs is its “view".
4. Trivial to achieve for $\mathcal{L} \in \mathcal{BPP}$
5. Extension: auxiliary input
6. The “standard" $\mathcal{NP}$ proof is typically not zero knowledge

Zero knowledge Proof

Definition 7 (zero-knowledge proofs)

An interactive proof (P,V) is computational zero-knowledge proof ($\mathcal{CZK}$) for $\mathcal{L}$, if $\forall$ PPT V^*, $\exists$ PPT S such that $\{\langle(\mathsf{P},\mathsf{V}^*)(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$.

Perfect $\mathcal{ZK}$ ($\mathcal{PZK}$)/statistical $\mathcal{ZK}$ ($\mathcal{SZK}$) — the above distributions are identicallly/statistically close.

1. $\mathcal{ZK}$ is a property of the prover.
2. $\mathcal{ZK}$ only required to hold with respect to true statements.
3. wlg. V^*'s outputs is its “view".
4. Trivial to achieve for $\mathcal{L} \in \mathcal{BPP}$
5. Extension: auxiliary input
6. The “standard" $\mathcal{NP}$ proof is typically not zero knowledge
7. Next class — $\mathcal{ZK}$ for all $\mathcal{NP}$

Section 2

Zero-Knowledge Proof for Graph Isomorphism

$\mathcal{ZK}$ Proof for Graph Isomorphism

Idea: route finding

$\mathcal{ZK}$ Proof for Graph Isomorphism

Idea: route finding

Protocol 8 ((P, V))

Common input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$

P's input: a permutation π over $[m]$ such that $\pi(E_1) = E_0$.

1. P chooses $\pi' \leftarrow \Pi_m$ and sends $E = \pi'(E_0)$ to V.
2. V sends $b \leftarrow \{0, 1\}$ to P.
3. If $b = 0$, P sets $\pi'' = \pi'$, otherwise, it sends $\pi'' = \pi' \circ \pi$ to V.
4. V accepts iff $\pi''(E_b) = E$.

$\mathcal{ZK}$ Proof for Graph Isomorphism

Idea: route finding

Protocol 8 ((P, V))

Common input: $x = (G_0 = ([m], E_0), G_1 = ([m], E_1))$

P's input: a permutation π over $[m]$ such that $\pi(E_1) = E_0$.

1. P chooses $\pi' \leftarrow \Pi_m$ and sends $E = \pi'(E_0)$ to V.
2. V sends $b \leftarrow \{0, 1\}$ to P.
3. If $b = 0$, P sets $\pi'' = \pi'$, otherwise, it sends $\pi'' = \pi' \circ \pi$ to V.
4. V accepts iff $\pi''(E_b) = E$.

Claim 9

Protocol 8 is a $\mathcal{SZK}$ for $\mathcal{GI}$, with perfect completeness and soundness $\frac{1}{2}$.

Proving Claim 9

- Completeness: Clear

Proving Claim 9

- Completeness: Clear
- Soundness: If exist $j \in \{0,1\}$ for which $\nexists \pi' \in \Pi_m$ with $\pi'(E_j) = E$, then V rejects w.p. at least $\frac{1}{2}$.

  Assuming V rejects w.p. less than $\frac{1}{2}$ and let π_0 and π_1 be the values guaranteed by the above observation (i.e., mapping E_0 and E_1 to E respectively).

  Then $\pi_0^{-1}(\pi_1(E_1)) = \pi_0 \implies (\mathsf{G}_0, \mathsf{G}_1) \in \mathcal{GI}$.

Proving Claim 9

- Completeness: Clear
- Soundness: If exist $j \in \{0,1\}$ for which $\nexists \pi' \in \Pi_m$ with $\pi'(E_j) = E$, then V rejects w.p. at least $\frac{1}{2}$.

  Assuming V rejects w.p. less than $\frac{1}{2}$ and let π_0 and π_1 be the values guaranteed by the above observation (i.e., mapping E_0 and E_1 to E respectively).

  Then $\pi_0^{-1}(\pi_1(E_1)) = \pi_0 \implies (\mathsf{G}_0, \mathsf{G}_1) \in \mathcal{GI}$.
- $\mathcal{ZK}$: Idea – for $(\mathsf{G}_0, \mathsf{G}_1) \in \mathcal{GI}$, it is easy to generate a random transcript for Steps 1-2, and to be able to open it with prob $\frac{1}{2}$.

The simulator

For a start consider a deterministic cheating verifier V^* that never aborts.

The simulator

For a start consider a deterministic cheating verifier V^* that never aborts.

Algorithm 10 (S)

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$

Do $|x|$ times:

1. Choose $b' \leftarrow \{0,1\}$ and $\pi \leftarrow \Pi_m$, and “send" $\pi(E_{b'})$ to $\mathsf{V}^*(x)$.
2. Let b be V^*'s answer. If $b = b'$, send π to V^*, output V^*'s output and halt. Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

The simulator

For a start consider a deterministic cheating verifier V^* that never aborts.

Algorithm 10 (S)

Input: $x = (G_0 = ([m], E_0), G_1 = ([m], E_1))$

Do $|x|$ times:

1. Choose $b' \leftarrow \{0, 1\}$ and $\pi \leftarrow \Pi_m$, and “send" $\pi(E_{b'})$ to $V^*(x)$.
2. Let b be V^*'s answer. If $b = b'$, send π to V^*, output V^*'s output and halt. Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Claim 11

$\{\langle (P, V^*)(x)\rangle\}_{x \in \mathcal{GI}} \approx \{S(x)\}_{x \in \mathcal{GI}}$

where $\langle (P, V^*)(x)\rangle$ describes the parties' join output in a random execution of (P, V^*) on input x.

The simulator

For a start consider a deterministic cheating verifier V^* that never aborts.

Algorithm 10 (S)

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$

Do $|x|$ times:

1. Choose $b' \leftarrow \{0,1\}$ and $\pi \leftarrow \Pi_m$, and "send" $\pi(E_{b'})$ to $\mathsf{V}^*(x)$.
2. Let b be V^*'s answer. If $b = b'$, send π to V^*, output V^*'s output and halt. Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Claim 11

$\{\langle(\mathsf{P}, \mathsf{V}^*)(x)\rangle\}_{x \in \mathcal{GI}} \approx \{\mathsf{S}(x)\}_{x \in \mathcal{GI}}$

where $\langle(\mathsf{P}, \mathsf{V}^*)(x)\rangle$ describes the parties' join output in a random execution of $(\mathsf{P}, \mathsf{V}^*)$ on input x.

Claim 11 implies that Protocol 8 is zero knowledge. (?)

Proving Claim 11

Consider the following inefficient simulator:

Algorithm 12 (S')

Input: $x = (G_0 = ([m], E_0), G_1 = ([m], E_1))$.

Do $|x|$ times:

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $V^*(x)$.
2. Let b be V^*'s answer.

   W.p. $\frac{1}{2}$,

   1. Find π' such that $E = \pi'(E_b)$, and send it to V^*.
   2. Output V^*'s output and halt.

   Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Proving Claim 11

Consider the following inefficient simulator:

Algorithm 12 (S′)

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$.

Do $|x|$ times:

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $\mathsf{V}^*(x)$.
2. Let b be V^*'s answer.

   W.p. $\frac{1}{2}$,

   1. Find π' such that $E = \pi'(E_b)$, and send it to V^*.
   2. Output V^*'s output and halt.

   Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Claim 13

$\mathsf{S}(x) \equiv \mathsf{S}'(x)$ for any $x \in \mathcal{GI}$.

Proving Claim 11

Consider the following inefficient simulator:

Algorithm 12 (S′)

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$.

Do $|x|$ times:

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $\mathsf{V}^*(x)$.
2. Let b be V^*'s answer.

   W.p. $\frac{1}{2}$,

   1. Find π' such that $E = \pi'(E_b)$, and send it to V^*.
   2. Output V^*'s output and halt.

   Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Claim 13

$\mathsf{S}(x) \equiv \mathsf{S}'(x)$ for any $x \in \mathcal{GI}$.

Proof: ?

Proving Claim 11 cont.

Consider a second inefficient simulator:

Algorithm 14 (S'')

Input: $x = (G_0 = ([m], E_0), G_1 = ([m], E_1))$

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $V^*(x)$.
2. Find π' such that $E = \pi'(E_b)$ and send it to V^*
3. Output V^*'s output and halt.

Proving Claim 11 cont.

Consider a second inefficient simulator:

Algorithm 14 (S'')

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $\mathsf{V}^*(x)$.
2. Find π' such that $E = \pi'(E_b)$ and send it to V^*
3. Output V^*'s output and halt.

Claim 15

$\forall x \in \mathcal{GI}$ it holds that

1. $\langle(\mathsf{P}, \mathsf{V}^*(x))\rangle \equiv \mathsf{S}''(x)$.

Proving Claim 11 cont.

Consider a second inefficient simulator:

Algorithm 14 (S'')

Input: $x = (G_0 = ([m], E_0), G_1 = ([m], E_1))$

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $V^*(x)$.
2. Find π' such that $E = \pi'(E_b)$ and send it to V^*
3. Output V^*'s output and halt.

Claim 15

$\forall x \in \mathcal{GI}$ it holds that

1. $\langle (P, V^*(x)) \rangle \equiv S''(x)$.
2. $SD(S''(x), S'(x)) \leq 2^{-|x|}$.

Proving Claim 11 cont.

Consider a second inefficient simulator:

Algorithm 14 (S'')

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $\mathsf{V}^*(x)$.
2. Find π' such that $E = \pi'(E_b)$ and send it to V^*
3. Output V^*'s output and halt.

Claim 15

$\forall x \in \mathcal{GI}$ it holds that

1. $\langle(\mathsf{P}, \mathsf{V}^*(x))\rangle \equiv \mathsf{S}''(x)$.
2. $\mathsf{SD}(\mathsf{S}''(x), \mathsf{S}'(x)) \leq 2^{-|x|}$.

Proving Claim 11 cont.

Consider a second inefficient simulator:

Algorithm 14 (S'')

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $\mathsf{V}^*(x)$.
2. Find π' such that $E = \pi'(E_b)$ and send it to V^*
3. Output V^*'s output and halt.

Claim 15

$\forall x \in \mathcal{GI}$ it holds that

1. $\langle(\mathsf{P}, \mathsf{V}^*(x))\rangle \equiv \mathsf{S}''(x)$.
2. $\mathsf{SD}(\mathsf{S}''(x), \mathsf{S}'(x)) \leq 2^{-|x|}$.

Proof: ?

Proving Claim 11 cont.

Consider a second inefficient simulator:

Algorithm 14 (S'')

Input: $x = (\mathsf{G}_0 = ([m], E_0), \mathsf{G}_1 = ([m], E_1))$

1. Choose $\pi \leftarrow \Pi_m$ and send $E = \pi(E_0)$ to $\mathsf{V}^*(x)$.
2. Find π' such that $E = \pi'(E_b)$ and send it to V^*
3. Output V^*'s output and halt.

Claim 15

$\forall x \in \mathcal{GI}$ it holds that

1. $\langle(\mathsf{P}, \mathsf{V}^*(x))\rangle \equiv \mathsf{S}''(x)$.
2. $\mathsf{SD}(\mathsf{S}''(x), \mathsf{S}'(x)) \leq 2^{-|x|}$.

Proof: ? (1) is clear.

Proving Claim 15(2)

Fix (E, π') and let $\alpha = \Pr_{S''(x)}[(E, \pi')]$.

Proving Claim 15(2)

Fix (E, π') and let $\alpha = \Pr_{S''(x)}[(E, \pi')]$.
It holds that

$$\Pr_{S'(x)}[(E, \pi')] = \alpha \cdot \sum_{i=1}^{|x|} (1 - \frac{1}{2})^{i-1} \cdot \frac{1}{2}$$
$$= (1 - 2^{-|x|}) \cdot \alpha$$

Proving Claim 15(2)

Fix (E, π') and let $\alpha = \Pr_{S''(x)}[(E, \pi')]$.
It holds that

$$\Pr_{S'(x)}[(E, \pi')] = \alpha \cdot \sum_{i=1}^{|x|} (1 - \frac{1}{2})^{i-1} \cdot \frac{1}{2}$$
$$= (1 - 2^{-|x|}) \cdot \alpha$$

Hence, $SD(S''(x), S'(x)) \leq 2^{-|x|} \Box$

Remarks

1. Randomized verifiers

Remarks

1. Randomized verifiers
2. Aborting verifiers

Remarks

1. Randomized verifiers
2. Aborting verifiers
3. Auxiliary input

Remarks

1. Randomized verifiers
2. Aborting verifiers
3. Auxiliary input
4. Negligible soundness error?

Remarks

1. Randomized verifiers
2. Aborting verifiers
3. Auxiliary input
4. Negligible soundness error?

Remarks

1. Randomized verifiers
2. Aborting verifiers
3. Auxiliary input
4. Negligible soundness error?

   Sequential repetition

Remarks

1. Randomized verifiers
2. Aborting verifiers
3. Auxiliary input
4. Negligible soundness error?

   Sequential repetition

   Parallel repetition

Remarks

1. Randomized verifiers
2. Aborting verifiers
3. Auxiliary input
4. Negligible soundness error?

   Sequential repetition

   Parallel repetition
5. Perfect $\mathcal{ZK}$ for “expected time simulators"

Remarks

1. Randomized verifiers
2. Aborting verifiers
3. Auxiliary input
4. Negligible soundness error?

   Sequential repetition

   Parallel repetition
5. Perfect $\mathcal{ZK}$ for “expected time simulators"
6. “Black box" simulation

"Transcript simulation" might not suffice!

"Transcript simulation" might not suffice!

Let $(\mathsf{G},\mathsf{E},\mathsf{D})$ be a public-key encryption scheme and let $\mathcal{L} \in \mathcal{NP}$.

"Transcript simulation" might not suffice!

Let $(\mathsf{G}, \mathsf{E}, \mathsf{D})$ be a public-key encryption scheme and let $\mathcal{L} \in \mathcal{NP}$.

Protocol 16 ((P, V))

Common input: $x \in \{0,1\}^*$

P's input: $w \in R_{\mathcal{L}}(x)$

1. V chooses $(d, e) \leftarrow \mathsf{G}(1^{|x|})$ and sends e to P
2. P sends $c = \mathsf{E}_e(w)$ to V
3. V accepts iff $\mathsf{D}_d(c) \in R_{\mathcal{L}}(x)$

"Transcript simulation" might not suffice!

Let $(\mathsf{G}, \mathsf{E}, \mathsf{D})$ be a public-key encryption scheme and let $\mathcal{L} \in \mathcal{NP}$.

Protocol 16 ((P, V))

Common input: $x \in \{0,1\}^*$

P's input: $w \in R_{\mathcal{L}}(x)$

1. V chooses $(d, e) \leftarrow \mathsf{G}(1^{|x|})$ and sends e to P
2. P sends $c = \mathsf{E}_e(w)$ to V
3. V accepts iff $\mathsf{D}_d(c) \in R_{\mathcal{L}}(x)$

- The above protocol has perfect completeness and soundness.

"Transcript simulation" might not suffice!

Let $(\mathsf{G}, \mathsf{E}, \mathsf{D})$ be a public-key encryption scheme and let $\mathcal{L} \in \mathcal{NP}$.

Protocol 16 ((P, V))

Common input: $x \in \{0,1\}^*$

P's input: $w \in R_{\mathcal{L}}(x)$

1. V chooses $(d, e) \leftarrow \mathsf{G}(1^{|x|})$ and sends e to P
2. P sends $c = \mathsf{E}_e(w)$ to V
3. V accepts iff $\mathsf{D}_d(c) \in R_{\mathcal{L}}(x)$

- The above protocol has perfect completeness and soundness.
- Is it zero-knowledge?

"Transcript simulation" might not suffice!

Let $(\mathsf{G}, \mathsf{E}, \mathsf{D})$ be a public-key encryption scheme and let $\mathcal{L} \in \mathcal{NP}$.

Protocol 16 ((P, V))

Common input: $x \in \{0,1\}^*$

P's input: $w \in R_{\mathcal{L}}(x)$

1. V chooses $(d, e) \leftarrow \mathsf{G}(1^{|x|})$ and sends e to P
2. P sends $c = \mathsf{E}_e(w)$ to V
3. V accepts iff $\mathsf{D}_d(c) \in R_{\mathcal{L}}(x)$

- The above protocol has perfect completeness and soundness.
- Is it zero-knowledge?
- It has "transcript simulator" (at least for honest verifiers): exits PPT S such that $\{\langle(\mathsf{P}(w \in R_{\mathcal{L}}(x)), \mathsf{V})(x)\rangle\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}(x)\}_{x\in\mathcal{L}}$,

  where *trans* stands for the transcript of the protocol (i.e., the messages exchange through the execution).

Section 3

Black-box Zero Knowledge

Black-box simulators

Definition 17 (Black-box simulator)

(P,V) is $\mathcal{CZK}$ with black-box simulation for $\mathcal{L}$, if $\exists$ oracle-aided PPT S s.t.

$$\{(\mathsf{P}(w_x),\mathsf{V}^*(z_x))(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}^{\mathsf{V}^*(x,z_x)}(x)\}_{x\in\mathcal{L}}$$

for any deterministic polynomial-time[a] V^* and $\{(w_x,z_x)\in R_{\mathcal{L}}(x)\times\{0,1\}^*\}_{x\in\mathcal{L}}$.

Prefect and statistical variants are defined analogously.

[a]Length of auxiliary input does not count for the running time.

Black-box simulators

Definition 17 (Black-box simulator)

(P,V) is $\mathcal{CZK}$ with black-box simulation for $\mathcal{L}$, if $\exists$ oracle-aided PPT S s.t.

$$\{(\mathsf{P}(w_x),\mathsf{V}^*(z_x))(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}^{\mathsf{V}^*(x,z_x)}(x)\}_{x\in\mathcal{L}}$$

for any deterministic polynomial-time[a] V^* and $\{(w_x,z_x)\in R_{\mathcal{L}}(x)\times\{0,1\}^*\}_{x\in\mathcal{L}}$.

Prefect and statistical variants are defined analogously.

[a] Length of auxiliary input does not count for the running time.

1. What about randomized verifier?

Black-box simulators

Definition 17 (Black-box simulator)

(P,V) is $\mathcal{CZK}$ with black-box simulation for $\mathcal{L}$, if $\exists$ oracle-aided PPT S s.t.

$$\{(\mathsf{P}(w_x),\mathsf{V}^*(z_x))(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}^{\mathsf{V}^*(x,z_x)}(x)\}_{x\in\mathcal{L}}$$

for any deterministic polynomial-time[a] V^* and $\{(w_x,z_x)\in R_{\mathcal{L}}(x)\times\{0,1\}^*\}_{x\in\mathcal{L}}$.

Prefect and statistical variants are defined analogously.

[a] Length of auxiliary input does not count for the running time.

1. What about randomized verifier?
2. “Most simulators" are black box

Black-box simulators

Definition 17 (Black-box simulator)

(P,V) is $\mathcal{CZK}$ with black-box simulation for $\mathcal{L}$, if $\exists$ oracle-aided PPT S s.t.

$$\{(\mathsf{P}(w_x),\mathsf{V}^*(z_x))(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}^{\mathsf{V}^*(x,z_x)}(x)\}_{x\in\mathcal{L}}$$

for any deterministic polynomial-time[a] V^* and $\{(w_x,z_x)\in R_{\mathcal{L}}(x)\times\{0,1\}^*\}_{x\in\mathcal{L}}$.

Prefect and statistical variants are defined analogously.

[a] Length of auxiliary input does not count for the running time.

1. What about randomized verifier?
2. “Most simulators" are black box

Black-box simulators

Definition 17 (Black-box simulator)

(P,V) is $\mathcal{CZK}$ with black-box simulation for $\mathcal{L}$, if $\exists$ oracle-aided PPT S s.t.

$$\{(\mathsf{P}(w_x),\mathsf{V}^*(z_x))(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}^{\mathsf{V}^*(x,z_x)}(x)\}_{x\in\mathcal{L}}$$

for any deterministic polynomial-time[a] V^* and $\{(w_x,z_x)\in R_{\mathcal{L}}(x)\times\{0,1\}^*\}_{x\in\mathcal{L}}$.

Prefect and statistical variants are defined analogously.

[a] Length of auxiliary input does not count for the running time.

1. What about randomized verifier?
2. “Most simulators" are black box
3. Strictly weaker then general simulation!

Section 4

Zero Knowledge for all NP

$\mathcal{CZK}$ for 3COL

- Assuming that OWFs exists, we give a (black-box) $\mathcal{CZK}$ for 3COL .

$\mathcal{CZK}$ for 3COL

- Assuming that OWFs exists, we give a (black-box) $\mathcal{CZK}$ for 3COL .
- We show how to transform it for any $\mathcal{L} \in \mathcal{NP}$ (using that 3COL $\in \mathcal{NPC}$).

$\mathcal{CZK}$ for 3COL

- Assuming that OWFs exists, we give a (black-box) $\mathcal{CZK}$ for 3COL .
- We show how to transform it for any $\mathcal{L} \in \mathcal{NP}$ (using that 3COL $\in \mathcal{NPC}$).

$\mathcal{CZK}$ for 3COL

- Assuming that OWFs exists, we give a (black-box) $\mathcal{CZK}$ for 3COL .
- We show how to transform it for any $\mathcal{L} \in \mathcal{NP}$ (using that 3COL $\in \mathcal{NPC}$).

Definition 18 (3COL)

$G = (M, E) \in$ 3COL, if $\exists\ \phi \colon M \mapsto [3]$ s.t. $\phi(u) \neq \phi(v)$ for every $(u, v) \in E$.

$\mathcal{CZK}$ for 3COL

- Assuming that OWFs exists, we give a (black-box) $\mathcal{CZK}$ for 3COL .
- We show how to transform it for any $\mathcal{L} \in \mathcal{NP}$ (using that 3COL $\in \mathcal{NPC}$).

Definition 18 (3COL)

$G = (M, E) \in$ 3COL, if $\exists\ \phi \colon M \mapsto [3]$ s.t. $\phi(u) \neq \phi(v)$ for every $(u, v) \in E$.

We use commitment schemes.

The protocol

Let π_3 be the set of all permutations over $[3]$.

The protocol

Let π_3 be the set of all permutations over $[3]$. We use perfectly binding commitment $\mathsf{Com} = (\mathsf{Snd}, \mathsf{Rcv})$.

The protocol

Let π_3 be the set of all permutations over [3]. We use perfectly binding commitment $\mathsf{Com} = (\mathsf{Snd}, \mathsf{Rcv})$.

Protocol 19 ((P, V))

Common input: Graph $\mathsf{G} = (M, E)$ with $n = |\mathsf{G}|$

P's input: a (valid) coloring ϕ of G

1. P chooses $\pi \leftarrow \Pi_3$ and sets $\psi = \pi \circ \phi$
2. $\forall v \in M$: P commits to $\psi(v)$ using Com (with security parameter 1^n).
   Let c_v and d_v be the resulting commitment and decommitment.
3. V sends $e = (u, v) \leftarrow E$ to P
4. P sends $(d_u, \psi(u)), (d_v, \psi(v))$ to V
5. V verifies that
   1. Both decommitments are valid,
   2. $\psi(u), \psi(v) \in [3]$, and
   3. $\psi(u) \neq \psi(v)$.

Claim 20

The above protocol is a $\mathcal{CZK}$ for 3COL, with perfect completeness and soundness $1/|E|$.

Claim 20

The above protocol is a $\mathcal{CZK}$ for 3COL, with perfect completeness and soundness $1/|E|$.

- Completeness: Clear

Claim 20

The above protocol is a $\mathcal{CZK}$ for 3COL, with perfect completeness and soundness $1/|E|$.

- Completeness: Clear
- Soundness: Let $\{c_v\}_{v \in M}$ be the commitments resulting from an interaction of V with an arbitrary P*.

  Define $\phi\colon M \mapsto [3]$ as follows:

  $\forall v \in M$: let $\phi(v)$ be the (single) value that it is possible to decommit c_v into (if not in [3], set $\phi(v) = 1$).

Claim 20

The above protocol is a $\mathcal{CZK}$ for 3COL, with perfect completeness and soundness $1/|E|$.

- Completeness: Clear
- Soundness: Let $\{c_v\}_{v \in M}$ be the commitments resulting from an interaction of V with an arbitrary P*.

  Define $\phi\colon M \mapsto [3]$ as follows:

  $\forall v \in M$: let $\phi(v)$ be the (single) value that it is possible to decommit c_v into (if not in $[3]$, set $\phi(v) = 1$).

Claim 20

The above protocol is a $\mathcal{CZK}$ for 3COL, with perfect completeness and soundness $1/|E|$.

- Completeness: Clear
- Soundness: Let $\{c_v\}_{v \in M}$ be the commitments resulting from an interaction of V with an arbitrary P^*.

  Define $\phi\colon M \mapsto [3]$ as follows:

  $\forall v \in M$: let $\phi(v)$ be the (single) value that it is possible to decommit c_v into (if not in $[3]$, set $\phi(v) = 1$).

  If $G \notin 3COL$, then $\exists (u, v) \in E$ s.t. $\psi(u) = \psi(v)$.

Claim 20

The above protocol is a $\mathcal{CZK}$ for 3COL, with perfect completeness and soundness $1/|E|$.

- Completeness: Clear
- Soundness: Let $\{c_v\}_{v \in M}$ be the commitments resulting from an interaction of V with an arbitrary P^*.

  Define $\phi\colon M \mapsto [3]$ as follows:

  $\forall v \in M$: let $\phi(v)$ be the (single) value that it is possible to decommit c_v into (if not in $[3]$, set $\phi(v) = 1$).

  If $G \notin$ 3COL, then $\exists (u, v) \in E$ s.t. $\psi(u) = \psi(v)$.

  Hence V rejects such x w.p. a least $1/|E|$

Proving $\mathcal{ZK}$

Fix a deterministic, non-aborting V^* that gets no auxiliary input.

Proving $\mathcal{ZK}$

Fix a deterministic, non-aborting V^* that gets no auxiliary input.

Algorithm 21 (S)

Input: A graph $\mathsf{G} = (M, E)$ with $n = |\mathsf{G}|$

Do $n \cdot |E|$ times:

1. Choose $e' = (u, v) \leftarrow E$.
   1. Set $\psi(u) \leftarrow [3]$,
   2. Set $\psi(v) \leftarrow [3] \setminus \{\psi(u)\}$, and
   3. Set $\psi(w) = 1$ for $w \in M \setminus \{u, v\}$.
2. $\forall v \in M$: commit to $\psi(v)$ to V^* (resulting in c_v and d_v)
3. Let e be the edge sent by V^*.

   If $e = e'$, send $(d_u, \psi(u)), (d_v, \psi(v))$ to V^*, output V^*'s output and halt.

   Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Proving $\mathcal{ZK}$ cont.

Claim 22

$\{(\mathsf{P}(w_x), \mathsf{V}^*)(x)\}_{x \in 3\mathrm{COL}} \approx_c \{\mathsf{S}^{\mathsf{V}^*(x)}(x)\}_{x \in 3\mathrm{COL}}$, for any $\{w_x \in R_{3\mathrm{COL}}(x)\}_{x \in 3\mathrm{COL}}$.

Consider the following (inefficient simulator)

Algorithm 23 (S')

Input: $\mathsf{G} = (V, E)$ with $n = |\mathsf{G}|$

Find (using brute force) a valid coloring ϕ of G

Do for $n \cdot |E|$ times:

1. Act like the honest prover does given private input ϕ.
2. Let e be the edge sent by V^*. W.p. $1/|E|$,
   1. Send $(\psi(u), d_u), (\psi(v), d_v)$ to V^*,
   2. Output V^*'s output and halt.

   Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Consider the following (inefficient simulator)

Algorithm 23 (S')

Input: $\mathsf{G} = (V, E)$ with $n = |\mathsf{G}|$

Find (using brute force) a valid coloring ϕ of G

Do for $n \cdot |E|$ times:

1. Act like the honest prover does given private input ϕ.
2. Let e be the edge sent by V^*. W.p. $1/|E|$,
    1. Send $(\psi(u), d_u), (\psi(v), d_v)$ to V^*,
    2. Output V^*'s output and halt.

    Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Claim 24

$\{\mathsf{S}^{\mathsf{V}^*(x)}(x)\}_{x \in 3\mathrm{COL}} \approx_c \{\mathsf{S}'^{\mathsf{V}^*(x)}(x)\}_{x \in 3\mathrm{COL}}$

Consider the following (inefficient simulator)

Algorithm 23 (S')

Input: $G = (V, E)$ with $n = |G|$

Find (using brute force) a valid coloring ϕ of G

Do for $n \cdot |E|$ times:

1. Act like the honest prover does given private input ϕ.
2. Let e be the edge sent by V^*. W.p. $1/|E|$,
   1. Send $(\psi(u), d_u), (\psi(v), d_v)$ to V^*,
   2. Output V^*'s output and halt.

   Otherwise, rewind V^* to its initial step, and go to step 1.

Abort.

Claim 24

$\{S^{V^*(x)}(x)\}_{x \in 3COL} \approx_c \{S'^{V^*(x)}(x)\}_{x \in 3COL}$

Proof: ?

Proving Claim 24

Assume $\exists$ PPT D, $p \in$ poly and an infinite set $\mathcal{I} \subseteq$ 3COL s.t.

$$\left|\Pr[\mathsf{D}(|x|, \mathsf{S}^{\mathsf{V}^*(x)}(x)) = 1] - \Pr[\mathsf{D}(|x|, \mathsf{S'}^{\mathsf{V}^*(x)}(x)) = 1]\right| \geq 1/p(|x|)$$

for all $x \in \mathcal{I}$.

Proving Claim 24

Assume $\exists$ PPT D, $p \in$ poly and an infinite set $\mathcal{I} \subseteq$ 3COL s.t.

$$\left|\Pr[\mathsf{D}(|x|, \mathsf{S}^{\mathsf{V}^*(x)}(x)) = 1] - \Pr[\mathsf{D}(|x|, \mathsf{S'}^{\mathsf{V}^*(x)}(x)) = 1]\right| \geq 1/p(|x|)$$

for all $x \in \mathcal{I}$.

Hence, $\exists$ PPT R^* and $b \in [3] \setminus 1$ such that

$$\{\mathsf{View}_{\mathsf{R}^*}(\mathsf{Snd}(1), \mathsf{R}^*(x))(1^{|x|})\}_{x\in\mathcal{I}} \not\approx_c \{\mathsf{View}_{\mathsf{R}^*}(\mathsf{Snd}(b), \mathsf{R}^*(x))(1^{|x|})\}_{x\in\mathcal{I}}$$

Proving Claim 24

Assume $\exists$ PPT D, $p \in$ poly and an infinite set $\mathcal{I} \subseteq$ 3COL s.t.

$$\left|\Pr[\mathrm{D}(|x|, \mathrm{S}^{\mathrm{V}^*(x)}(x)) = 1] - \Pr[\mathrm{D}(|x|, \mathrm{S}'^{\mathrm{V}^*(x)}(x)) = 1]\right| \geq 1/p(|x|)$$

for all $x \in \mathcal{I}$.

Hence, $\exists$ PPT R^* and $b \in [3] \setminus 1$ such that

$$\{\mathrm{View}_{\mathrm{R}^*}(\mathrm{Snd}(1), \mathrm{R}^*(x))(1^{|x|})\}_{x \in \mathcal{I}} \not\approx_c \{\mathrm{View}_{\mathrm{R}^*}(\mathrm{Snd}(b), \mathrm{R}^*(x))(1^{|x|})\}_{x \in \mathcal{I}}$$

We critically used the non-uniform security of Com.

S' is a good simulator

Claim 25

$\{(\mathsf{P}(w_x), \mathsf{V}^*)(x)\}_{x \in 3\mathrm{COL}} \approx_c \{\mathsf{S}'^{\mathsf{V}^*(x)}(x)\}_{x \in 3\mathrm{COL}}$, for any $\{w_x \in R_{\mathcal{GI}}(x)\}_{x \in 3\mathrm{COL}}$.

S' is a good simulator

Claim 25

$\{(\mathsf{P}(w_x), \mathsf{V}^*)(x)\}_{x \in 3\mathrm{COL}} \approx_c \{\mathsf{S}'^{\mathsf{V}^*(x)}(x)\}_{x \in 3\mathrm{COL}}$, for any $\{w_x \in R_{\mathcal{GI}}(x)\}_{x \in 3\mathrm{COL}}$.

Proof: ?

Remarks

- Aborting verifiers

Remarks

- Aborting verifiers
- Auxiliary inputs

Remarks

- Aborting verifiers
- Auxiliary inputs
- Soundness amplification

Extending to all $\mathcal{NP}$

For $\mathcal{L} \in \mathcal{NP}$ let Map_X and Map_W be two poly-time functions s.t.

- $x \in \mathcal{L} \iff \mathsf{Map}_X(x) \in \mathsf{3COL}$,

Extending to all $\mathcal{NP}$

For $\mathcal{L} \in \mathcal{NP}$ let Map_X and Map_W be two poly-time functions s.t.

- $x \in \mathcal{L} \iff \mathsf{Map}_X(x) \in \mathsf{3COL}$,
- Map_X is efficiently invertible.

Extending to all $\mathcal{NP}$

For $\mathcal{L} \in \mathcal{NP}$ let Map_X and Map_W be two poly-time functions s.t.

- $x \in \mathcal{L} \iff \mathsf{Map}_X(x) \in \mathsf{3COL}$,
- Map_X is efficiently invertible.
- $(x, w) \in R_{\mathcal{L}} \iff \mathsf{Map}_W(x, w) \in R_{\mathsf{3COL}}(\mathsf{Map}_X(x))$

Extending to all $\mathcal{NP}$

For $\mathcal{L} \in \mathcal{NP}$ let Map_X and Map_W be two poly-time functions s.t.

- $x \in \mathcal{L} \iff \mathsf{Map}_X(x) \in \mathsf{3COL}$,
- Map_X is efficiently invertible.
- $(x, w) \in R_{\mathcal{L}} \iff \mathsf{Map}_W(x, w) \in R_{\mathsf{3COL}}(\mathsf{Map}_X(x))$

Extending to all $\mathcal{NP}$

For $\mathcal{L} \in \mathcal{NP}$ let Map_X and Map_W be two poly-time functions s.t.

- $x \in \mathcal{L} \iff \mathsf{Map}_X(x) \in \mathsf{3COL}$,
- Map_X is efficiently invertible.
- $(x, w) \in R_{\mathcal{L}} \iff \mathsf{Map}_W(x, w) \in R_{\mathsf{3COL}}(\mathsf{Map}_X(x))$

Let (P, V) be a $\mathcal{CZK}$ for $\mathsf{3COL}$.

Extending to all $\mathcal{NP}$

For $\mathcal{L} \in \mathcal{NP}$ let Map_X and Map_W be two poly-time functions s.t.

- $x \in \mathcal{L} \iff \mathsf{Map}_X(x) \in \mathsf{3COL}$,
- Map_X is efficiently invertible.
- $(x, w) \in R_{\mathcal{L}} \iff \mathsf{Map}_W(x, w) \in R_{\mathsf{3COL}}(\mathsf{Map}_X(x))$

Let (P, V) be a $\mathcal{CZK}$ for $\mathsf{3COL}$.

Protocol 26 ($(\mathsf{P}_{\mathcal{L}}, \mathsf{V}_{\mathcal{L}})$)

Common input: $x \in \{0, 1\}^*$.

$\mathsf{P}_{\mathcal{L}}$'s input: $w \in R_{\mathcal{L}}(x)$.

1. The two parties interact in $\langle(\mathsf{P}(\mathsf{Map}_W(x, w)), \mathsf{V})(\mathsf{Map}_X(x))\rangle$, where $\mathsf{P}_{\mathcal{L}}$ and $\mathsf{V}_{\mathcal{L}}$ taking the role of P and V respectively.
2. $\mathsf{V}_{\mathcal{L}}$ accepts iff V accepts in the above execution.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_{\mathcal{L}}, \mathsf{V}_{\mathcal{L}})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_{\mathcal{L}}, \mathsf{V}_{\mathcal{L}})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_{\mathcal{L}}, \mathsf{V}_{\mathcal{L}})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.
- Zero knowledge: Let S (an efficient) $\mathcal{ZK}$ simulator for (P, V) (for 3COL). Define $\mathsf{S}_{\mathcal{L}}(x)$ to output $\mathsf{S}(\mathsf{Map}_X(x))$, while replacing the string $\mathsf{Map}_X(x)$ in the output of S with x.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_{\mathcal{L}}, \mathsf{V}_{\mathcal{L}})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.
- Zero knowledge: Let S (an efficient) $\mathcal{ZK}$ simulator for (P, V) (for 3COL).

  Define $\mathsf{S}_{\mathcal{L}}(x)$ to output $\mathsf{S}(\mathsf{Map}_X(x))$, while replacing the string $\mathsf{Map}_X(x)$ in the output of S with x.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_\mathcal{L}, \mathsf{V}_\mathcal{L})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.
- Zero knowledge: Let S (an efficient) $\mathcal{ZK}$ simulator for (P, V) (for 3COL).

  Define $\mathsf{S}_\mathcal{L}(x)$ to output $\mathsf{S}(\mathsf{Map}_X(x))$, while replacing the string $\mathsf{Map}_X(x)$ in the output of S with x.

Claim 28

$\{(\mathsf{P}_\mathcal{L}(w_x), \mathsf{V}^*_\mathcal{L})(x)\}_{x \in \mathcal{L}} \approx_c \{\mathsf{S}_\mathcal{L}^{\mathsf{V}^*_\mathcal{L}(x)}(x)\}_{x \in \mathcal{L}}$ for any PPT $\mathsf{V}^*_\mathcal{L}$.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_{\mathcal{L}}, \mathsf{V}_{\mathcal{L}})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.
- Zero knowledge: Let S (an efficient) $\mathcal{ZK}$ simulator for (P, V) (for 3COL).
  Define $\mathsf{S}_{\mathcal{L}}(x)$ to output $\mathsf{S}(\mathsf{Map}_X(x))$, while replacing the string $\mathsf{Map}_X(x)$ in the output of S with x.

Claim 28

$\{(\mathsf{P}_{\mathcal{L}}(w_x), \mathsf{V}^*_{\mathcal{L}})(x)\}_{x \in \mathcal{L}} \approx_c \{\mathsf{S}_{\mathcal{L}}^{\mathsf{V}^*_{\mathcal{L}}(x)}(x)\}_{x \in \mathcal{L}}$ for any PPT $\mathsf{V}^*_{\mathcal{L}}$.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_\mathcal{L}, \mathsf{V}_\mathcal{L})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.
- Zero knowledge: Let S (an efficient) $\mathcal{ZK}$ simulator for (P, V) (for 3COL).

  Define $\mathsf{S}_\mathcal{L}(x)$ to output $\mathsf{S}(\mathsf{Map}_X(x))$, while replacing the string $\mathsf{Map}_X(x)$ in the output of S with x.

Claim 28

$\{(\mathsf{P}_\mathcal{L}(w_x), \mathsf{V}^*_\mathcal{L})(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}_\mathcal{L}^{\mathsf{V}^*_\mathcal{L}(x)}(x)\}_{x\in\mathcal{L}}$ for any PPT $\mathsf{V}^*_\mathcal{L}$.

Proof:

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_{\mathcal{L}}, \mathsf{V}_{\mathcal{L}})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.
- Zero knowledge: Let S (an efficient) $\mathcal{ZK}$ simulator for (P, V) (for 3COL).

  Define $\mathsf{S}_{\mathcal{L}}(x)$ to output $\mathsf{S}(\mathsf{Map}_X(x))$, while replacing the string $\mathsf{Map}_X(x)$ in the output of S with x.

Claim 28

$\{(\mathsf{P}_{\mathcal{L}}(w_x), \mathsf{V}^*_{\mathcal{L}})(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}_{\mathcal{L}}^{\mathsf{V}^*_{\mathcal{L}}(x)}(x)\}_{x\in\mathcal{L}}$ for any PPT $\mathsf{V}^*_{\mathcal{L}}$.

Proof: Assume $\{(\mathsf{P}_{\mathcal{L}}(w_x), \mathsf{V}^*_{\mathcal{L}})(x)\}_{x\in\mathcal{L}} \not\approx_c \{\mathsf{S}_{\mathcal{L}}^{\mathsf{V}^*_{\mathcal{L}}(x)}(x)\}_{x\in\mathcal{L}}$ for some $\mathsf{V}^*_{\mathcal{L}}$.

Extending to all $\mathcal{L} \in \mathcal{NP}$ cont.

Claim 27

$(\mathsf{P}_\mathcal{L}, \mathsf{V}_\mathcal{L})$ is a $\mathcal{CZK}$ for $\mathcal{L}$ with the same completeness and soundness as (P, V) as for 3COL.

- Completeness and soundness: Clear.
- Zero knowledge: Let S (an efficient) $\mathcal{ZK}$ simulator for (P, V) (for 3COL).

  Define $\mathsf{S}_\mathcal{L}(x)$ to output $\mathsf{S}(\mathsf{Map}_X(x))$, while replacing the string $\mathsf{Map}_X(x)$ in the output of S with x.

Claim 28

$\{(\mathsf{P}_\mathcal{L}(w_x), \mathsf{V}^*_\mathcal{L})(x)\}_{x\in\mathcal{L}} \approx_c \{\mathsf{S}_\mathcal{L}^{\mathsf{V}^*_\mathcal{L}(x)}(x)\}_{x\in\mathcal{L}}$ for any PPT $\mathsf{V}^*_\mathcal{L}$.

Proof: Assume $\{(\mathsf{P}_\mathcal{L}(w_x), \mathsf{V}^*_\mathcal{L})(x)\}_{x\in\mathcal{L}} \not\approx_c \{\mathsf{S}_\mathcal{L}^{\mathsf{V}^*_\mathcal{L}(x)}(x)\}_{x\in\mathcal{L}}$ for some $\mathsf{V}^*_\mathcal{L}$.

It follows that $\{(\mathsf{P}(\mathsf{Map}_W(x, w_x)), \mathsf{V}^*)(x)\}_{x\in 3\mathrm{COL}} \not\approx_c \{\mathsf{S}^{\mathsf{V}^*(x)}(x)\}_{x\in 3\mathrm{COL}}$.

$\mathsf{V}^*(x)$: act like $\mathsf{V}^*_\mathcal{L}(x')$ for $x' = \mathsf{Map}_X^{-1}(x)$.